

0967607-10000

UTILIZATION OF THIRD PARTY LEGACY DATA LIST

FIELD OF THE INVENTION

This invention relates generally to computer software, and more particularly to methods of allow use of legacy data lists by a newly installed application program.

BACKGROUND OF THE INVENTION

Data lists are used extensively in various application programs. For example, a phonebook is typically a requirement for computer facsimile software. As another example, e-mail address books and contact information lists are commonly incorporated into e-mail programs. Typically, the data list is maintained in a custom format for its particular application program, and not compatible with other application programs.

When a new application program is installed to replace an existing, or a legacy, application program, a new user must populate the data list associated with the new application program. Such population can be done manually or semi-automatically by use of an import function. In the latter case, a phonebook or address book, for example, from the legacy data list from the legacy application program is imported into the new application problem. A problem with either of the above technique is that two separate data lists must be maintained and synchronized. This requires extra and duplicated effort on the part of the user, and is thus undesirable.

Thus, there is a need for a method and apparatus for allowing a new application program to utilize data lists of one or more legacy application program without requiring a user to maintain and/or synchronize two separate data lists.

SUMMARY OF THE INVENTION

In accordance with the principles of the present invention, a method of providing an access to one or more third party legacy data list to a user of an application program of a computer system comprises querying an operating system, by the application program upon start of the application program, whether one or more plug-in module is registered in a registry of an operating system, the one or more plug-in modules being capable of interfacing

1 with corresponding respective ones of the one or more third party legacy data list, receiving,
2 from the one or more plug-in modules found in the registry, identifications of ones of the one
3 or more third party legacy data list corresponding to the found one or more plug-in modules,
4 and providing a list of the identifications to the user through a user interface of the application
5 program.

6 In accordance with another aspect of the present invention, a system for providing an
7 access to one or more third party legacy data list to a user of a computer system comprises
8 one or more plug-in module, each of which being capable of interfacing with an associated
9 one of the one or more third party data list, each of the one or more plug-in modules being
10 registered in a registry of an operating system of the computer system, and an application
11 program having a user interface, the application program upon starting being in
12 communication with the operating system to query the registry to determine registered ones
13 of the one or more plug-in module, the application program further configured to query each
14 of the registered ones of the one or more plug-in modules for names of the one or more third
15 party legacy data list, the application program being configured to provide a list of the names
16 of the one or more third party legacy data list to the user through the user interface.

17 In accordance with yet another aspect of the present invention, a computer readable
18 storage medium on which is embedded one or more computer programs, the one or more
19 computer programs implementing a method of providing an access to one or more third party
20 legacy data list to a user of an application program of a computer system, the one or more
21 computer programs comprises a set of instructions for querying an operating system, by the
22 application program upon start of the application program, whether one or more plug-in
23 module is registered in a registry of an operating system, the one or more plug-in modules
24 being capable of interfacing with corresponding respective ones of the one or more third party
25 legacy data list, receiving, from the one or more plug-in modules found in the registry,
26 identifications of ones of the one or more third party legacy data list corresponding to the
27 found one or more plug-in modules, and providing a list of the identifications to the user
28 through a user interface of the application program.

29 In comparison to known prior art, certain embodiments of the invention are capable of
30 achieving certain advantages, including some or all of the following: (1) saving time and

1 effort of the user; (2) eliminating the need for a user to populate a new data list; (3)
 2 eliminating the need to maintain and synchronize multiple data lists; (4) avoiding errors that
 3 can be introduced during the population process; and (5) eliminating the need for a user to
 4 learn a new user interface. Those skilled in the art will appreciate these and other advantages
 5 and benefits of various embodiments of the invention upon reading the following detailed
 6 description of a preferred embodiment with reference to the below-listed drawings.

8 BRIEF DESCRIPTION OF THE DRAWINGS

9 Features and advantages of the present invention will become apparent to those skilled
 10 in the art from the following description with reference to the drawings, in which:

11 Figure 1 shows an exemplary embodiment of the relevant portions of a system for
 12 providing user access of third party legacy data list in accordance with the principles of the
 13 present invention;

14 Figure 1A shows a more detailed depiction of an exemplary embodiment of one of the
 15 plug-in module shown in Figure 1;

16 Figure 2 shows a flowchart of an exemplary embodiment of the application program
 17 installation process in accordance with the principles of the present invention;

18 Figure 3 shows a flowchart of an exemplary embodiment of the plug-in modules
 19 discovery process in accordance with the principles of the present invention;

20 Figure 4 shows an illustrative exemplary embodiment of the user interface for the
 21 application program in accordance with the principles of the present invention;

22 Figure 5 shows a flowchart of an exemplary embodiment of the user selection and
 23 access of the third party legacy data list process in accordance with the principles of the
 24 present invention; and

25 Figure 6 shows a flowchart of an exemplary embodiment of the application update
 26 process in accordance with the principles of the present invention;

28 DETAILED DESCRIPTION OF A PREFERRED EMBODIMENT

29 For simplicity and illustrative purposes, the principles of the present invention are
 30 described by referring mainly to an exemplar embodiment thereof, particularly with

1 principles of the present invention. The system **100** comprises an operating system (OS) **101**
 2 installed on the computer (not shown). The OS **101** may be, for example, the WINDOWS™
 3 sold by the Microsoft Corporation of Redmond, Washington, USA. The operating system
 4 includes a component registry **102** that keeps a list of system components and resources
 5 required by an application program **103** installed on the computer. The application program
 6 **103** may comprise any program that is require to or is desirable to keep a data list. For
 7 example, a facsimile software may keep a list of names and facsimile telephone numbers of
 8 many potential recipients. An e-mail software may need to keep a list of e-mail addresses of
 9 various e-mail users. Many personal organizer programs keep contact information,
 10 scheduling/appointment lists, or the like.

11 The application program **103** may also comprise a user interface **104** for providing a
 12 user of the application program means for interact with the application program **103**. In
 13 accordance with the principles of the present invention, the user interface **104** serves as a
 14 common interface through which the user may access any number of third party legacy data
 15 lists **106**, i.e., the legacy data list #1 through legacy data list #N (where N may be any integer
 16 greater than equal to 1) installed on the computer.

17 The system **100** may also comprise any number of plug-in modules (PM) **105**, one of
 18 which is shown in more detail in Fig. 1A. As shown, each plug-in module **105** comprises an
 19 application program interface portion **105A** and a third party legacy (TPL) data list interface
 20 portion **105B**. The application program interface portions **105A** of all plug-in modules are
 21 identical to each other. Thus, the application program **103** may use one common protocol
 22 and/or function call set to communicate with all plug-in modules **105**. Thus, when an
 23 additional plug-in module is installed on the computer after the installation of the application
 24 program **103**, the application program **103** need not be recompiled or modified in any way.
 25 As will described in more detail later, the newly added plug-in modules are discovered by the
 26 application program **103**, and are used by the application in the same manner that the existing
 27 plug-in modules are used.

28 In accordance with an embodiment of the present invention, each of the plug-in
 29 modules **105** is compiled as a dynamic link library, with which the application program **103**
 30 may communicate during run-time. In a preferred embodiment of the present invention, the

1 the plug-in modules comprise adding a COM category entry in the registry, e.g., COM
2 category: <Legacy Data List Plug-in Modules>, and listing each of the plug-in modules 105
3 as a COM object with a unique identifier under that COM category.

4 Referring now to Figure 3, an exemplary embodiment of the plug-in modules
5 discovery process in accordance with the principles of the present invention will be described.

6 Upon installation and/or start of the application program, the application program in
7 step 301, the application program sends a standard Windows function call, e.g., a COM
8 category call, to query the operating system for the list of plug-in modules under the
9 particular COM category, e.g., the <Legacy Data List Plug-in Modules> COM category, in
10 step 302.

11 In step 303, the operating system 101 returns the unique identifiers of the plug-in
12 modules registered in the registry 102. The application program 103 at this point knows how
13 many plug-in modules are installed on the computer. In step 304, the application program,
14 using the unique identifiers, sends a name request function call to each of the plug-in
15 modules, and receives the name of the data list from each of the plug-in modules. The
16 application program 103, then sends an availability check function call to each of the plug-in
17 modules in step 305. The application program 103 makes a determination, in step 306,
18 whether any of the plug-in modules has returned with a response that indicates the
19 corresponding data list is installed on the computer. If no plug-in module indicates that the
20 data list it supports is installed on the computer, the application program 103 may ask the user
21 of the application program whether the user wishes to create a new data list in step 308, and if
22 the user so wishes, may provide a user interface screen (not shown) to allow the user to
23 manually enter new data to create a new data list in step 309. The application program 103
24 then waits for a user action in step 310.

25 If, on the other hand, it is determined (in step 306) that one or more data list is
26 installed on the computer, the application program 103 causes the names of all data lists that
27 are installed on the computer to be displayed to the user through the user interface 104 to
28 allow the user to select a desired data list from the listed data lists.

29 As can be appreciated from the above description, once a plug-in module is registered
30 with the operating system, the application program may discover the plug-in module and

1 program 103 in step 502. In step 504, the data set is then displayed to the user, e.g., in the
2 display window 402, from which the user can access the data.

3 If, in step 505, the application program 103 detects that the user wishes to modify any
4 datum in the selected third party legacy data list 105, e.g., when the "Edit" selection button
5 403 is selected, the application program 103 sends an edit request function call to the plug-in
6 module supporting the selected data list in step 506. The plug-in module then causes an edit
7 user interface screen (not shown) to be displayed in step 507, and the user is allowed to add,
8 delete and/or modify any datum from the data set of the selected data list. The edit user
9 interface screen may be either a user interface screen of the application program 103 or a user
10 interface screen of the legacy data list 106. The process then ends in step 509 until another
11 user action is detected in step 501, in which case, the process 500 is repeated.

12 Figure 6 shows a flowchart of an exemplary embodiment of the application update
13 process in accordance with the principles of the present invention. As previously mentioned,
14 even after the application program 103 is installed on a field computer, when a plug-in
15 module for a previously unsupported data list is desired, the plug-in may be developed and
16 delivered to the field, or downloaded from the Internet by the user, and installed on the field
17 computer.

18 The installer program for the new plug-in module may unpack and copy the new plug-
19 in module(s) to a memory storage of the computer, e.g., a hard disk, in step 602. In step 603,
20 the installer then registers the new plug-in module with the operating system in a similar
21 manner as previously described in connection with Fig. 2. Once the new plug-in module is
22 registered, the application program 103 will automatically discover the new plug-in module
23 the next time the application program is run as described above in Fig. 3.

24 What has been described and illustrated herein is a preferred embodiment of the
25 invention along with some of its variations. The terms, descriptions and figures used herein
26 are set forth by way of illustration only and are not meant as limitations. Those skilled in the
27 art will recognize that many variations are possible within the spirit and scope of the
28 invention, which is intended to be defined by the following claims -- and their equivalents --
29 in which all terms are meant in their broadest reasonable sense unless otherwise indicated.